Guide to running VASP

What is VASP

The Vienna Ab initio Simulation Package (VASP) is a computer program for atomic scale materials modelling, e.g. electronic structure calculations and quantum-mechanical molecular dynamics, from first principles.

VASP computes an approximate solution to the many-body Schrödinger equation, either within density functional theory (DFT), solving the Kohn-Sham equations, or within the Hartree-Fock (HF) approximation, solving the Roothaan equations. Hybrid functionals that mix the Hartree-Fock approach with density functional theory are implemented as well. Furthermore, Green's functions methods (GW quasiparticles, and ACFDT-RPA) and many-body perturbation theory (2nd-order Møller-Plesset) are available in VASP.

In VASP, central quantities like the one-electron orbitals, the electronic charge density, and the local potential are expressed in plane wave basis sets. The interactions between the electrons and ions are described using norm-conserving or ultrasoft pseudopotentials, or the projector-augmented-wave method.

Please note that only the parallel version of VASP is installed on HPC Wales, with the parallel executable named *vasp.* HPC Wales can make this application available to selected users, as a VASP licence is required. Please contact the HPCWales helpdesk to request access.

Step 1 - Log in

The example used in this guide is configured to run on the Swansea Sandy Bridge cluster. Connect to *login.hpcwales.co.uk* with your HPC Wales user credentials using your preferred method (e.g. PuTTY from a Windows machine or ssh from any Linux terminal), then **`ssh sw-sb-log-001`** to connect to the Swansea system.

The steps below involve typing commands (**`in bold font`**) in the terminal window.

Step 2 - Load a VASP module

A number of VASP packages are available.

- List pre-installed VASP versions:

```
module avail vasp
```

To load the latest version:

```
module purge
module load vasp/5.3.3
```

Note that successful execution of the associated executable relies on access to the MPI library specified by the module (Version 4.0 Update 3).

Ewrop & Chymru: Buddsoddi yn eich dyfodol
Cronfa Datblygu Rhanbarthol Ewrop
Europe & Wales: Investing in your future
European Regional Development Fund
ERDF
Llywodraeth Cymru
Welsh Government

- Confirm the loaded modules. All dependencies are handled automatically via the module file:

  ```
  module list
  ```

Step 3 - Create a directory

From your home directory, create a directory to hold the VASP data:

```
cd ~
mkdir VASP
```

Step 4 - Obtain a test case

Three benchmark test cases are provided with the installation at

```
/app/chemistry/vasp/5.3.3/sb-example
```

Copy the simplest of these contained in the tar file VaspHg.tar.gz to your user space:

```
cd ~/VASP
cp /app/chemistry/vasp/5.3.3/sb-example/VaspHg.tar.gz .
tar xvzf VaspHg.tar.gz
cd VaspHg
```

Note the full stop at the end of the cp line as this is important!

Step 5 - Submit a job

You are now ready to run this test case with the supplied SLURM job script.

The current directory should contain all required files to run a VASP job: INCAR is the central input file that determines ‘what to do and how', POTCAR contains the pseudopotential for each atomic species used in the calculation, POSCAR contains the lattice geometry and the ionic positions, and optionally also starting velocities and predictor-corrector coordinates for a MD-run, KPOINTS contains the k-point coordinates and weights of the mesh-size for creating the k-point grid. The directory also contains a sample output file and a batch script.

- Submit the job using: `sbatch VaspHg.SLURM.q`
- Check the job queue using: `squeue`
- When completed, the new output can be found in
  VASP.Hg.OUTCAR.NCPUS=32.PPN=16.<*Job_ID*>.

  where <*Job_ID*> is the ID generated by the system.
- The output should be compared with the reference output file OUTCAR.ref to check that all is working correctly.

Step 6 – Scaling tests

Now you have checked the code is working look at the scaling of the code. To achieve this run the job a number of times on differing core counts and calculate the speed up of the job relative to running on 16 cores. Remember if t(P) is the time on P cores the speed up S is given by S=t(16)/t(P). To do this

- Look at the batch script *VaspHg.SLURM.q* and you will see a line similar to #SBATCH --ntasks=32
- Using your favourite editor change the number at the end of this line to adjust the number of cores
- Re-run the job
- The elapsed time for the job can be found near the end of the OUTCAR file after the text string 'Elapsed time (sec)'.

Run the job on 8, 16, 24 and 32 cores and plot the speed up. What would you conclude about the scaling of the job?

Note: For larger number of cores it is important to find an appropriate value for the NPAR switch, which sets the parallelization over bands.

References

- Official VASP website: https://www.vasp.at/
- VASP User Documentation can be found at https://www.vasp.at/index.php/documentation

  with an on-line user manual at http://cms.mpi.univie.ac.at/vasp/vasp/vasp.html and a PDF copy at:

  http://cms.mpi.univie.ac.at/vasp/vasp.pdf.

The lecture notes and examples from a VASP hands on workshop (see https://www.vasp.at/index.php/documentation) are highly recommended as place to start if you are a beginner and might also be useful if you are not.

Ewrop & Chymru: Buddsoddi yn eich dyfodol
Cronfa Datblygu Rhanbarthol Ewrop
Europe & Wales: Investing in your future
European Regional Development Fund
ERDF
Llywodraeth Cymru Welsh Government